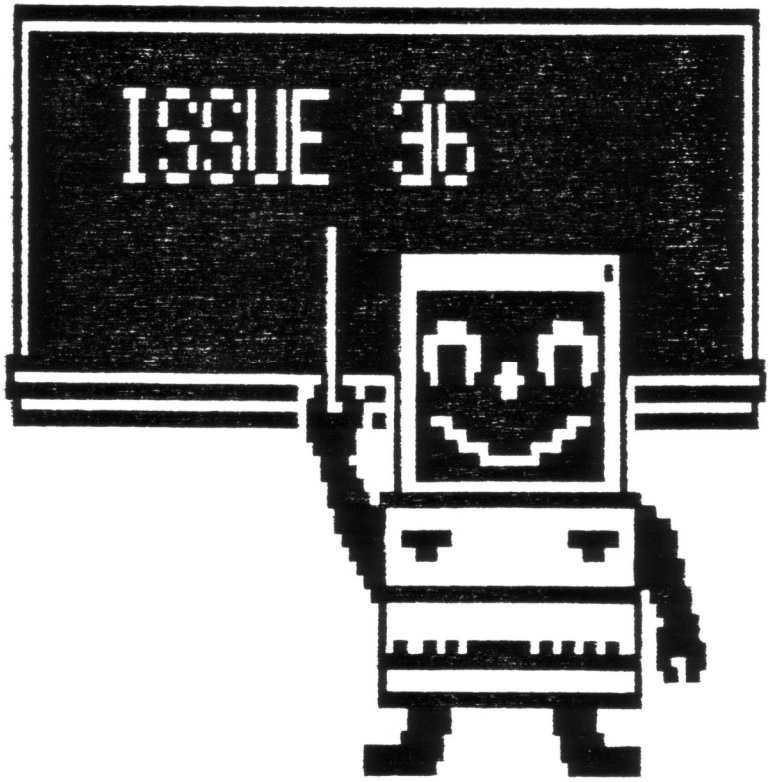
VZ-LINK MAGAZINE

NEW ZEALAND

VZ 300/200

COMPUTER USERS CLUB



OCTOBER 89

COMMENT

Welcome to the October issue.

Once again we have managed to rattle out another issue of Your Newsletter (even if it is late,again) Sorry.

Well,the gremlin's have hit my VZ speech synthesizer,with a error in the P.C.Board being the problem.
I hope to have this project up and running in time for the December meeting.
Now where have I heard that before.

Our membership now appears to be stable at around the 35 mark and anyone who has not paid their subs had better be quick,
as this is probably the last newsletter you will get mailed to you otherwise.

I would like to welcome our newest member, Lou Ellmers from Paihia.
I hope that we can help you and look forward to hearing from you,and maybe seeing you at a meeting sometime.

This issue has a new series called Micro Magic.
This is a true beginners guide to the VZ which was written and distributed by Gordon Browell of Australia and reprinted with his permission.
I hope it will help the newer members to understand programs a little better.

The last club meeting will be held on saturday 2nd December (details below).
I hope you all can make it,as it will be a super end of year affair.

see you next issue
Regards peter

****
AUCKLAND VZ 300/200 COMPUTER USERS CLUB
****

VZ USERS-WORKSHOP/MEETING

WE WILL BE HOLDING A VZ USERS WORKSHOP ON:-

SATURDAY 2 DECEMBER 89
STARTING AT 10-30 AM

AT THE :-

OWAIRAKA BAPTIST CHURCH
119 RICHARDSON ROAD
OWAIRAKA

BRING YOUR FAVOURITE SOFTWARE ETC
BRING YOUR PROBLEM'S

A LIGHT LUNCH WILL BE PROVIDED,BUT PLEASE BRING A PLATE TO HELP

ALFA TIMER BY STEVEN DILLON

```
1 'FROM SINCLAIR PROGRAMS FOR ZX81
2 'ADAPTED TO A VZ200 BY STEVEN DILLON
5 GOSUB300
10 H=400
20 REM
30 CLS:T=0:H=H
40 PRINT@15,"GO"
50 FORA=65TO90
60 IFINKEY$<>CHR$(A)THENGOTO200
70 PRINTCHR$(A);
75 SOUNDA-60,1
80 NEXT
90 PRINT@96,"YOU TOOK ";T;" TIME UNITS"
100 IFT<HTHENH=T
110 PRINT,,"HIGH=";H
120 PRINT,,,"HIT (S) TO TRY AGAIN"
130 IFINKEY$<>"S"THENGOTO130
140 T=0:GOTO20
200 T=T+1
210 GOTO60
300 CLS:PRINT:PRINT"            ALFA TIMER"
310 PRINT:PRINT
320 PRINT" THE OBJECT OF THIS GAME IS        TO SEE HOW QUICKLY"
340 PRINT" YOU CAN TYPE THE ALPHABET         FROM A TO Z"
350 FORW=0TO5000:NEXT:RETURN
```

INSIDE THE VZ

This month marks the start of our look at IO (input/output) Memory. This is the section of memory which is used to read from input devices such as the keyboard and cassette port, and write to output devices such as the VDU (Video Display Unit) and cassette output port. This month we're discussing the keyboard and how it's interfaced to the CPU.

The entire keyboard is arranged in a matrix. The arrangement is shown below. When any key is pressed, it links the column line with the respective row line. If you're not interested in the electronics of how the keyboard operates, skip the next few paragraphs.

The six column lines are connected to the data bus, via an octal buffer. This buffer is gated, meaning that the column lines are only electrically connected to the data bus when a logic low signal appears at the $\overline{IOR}$ pin. This logic Ø signal is sent by the addressing circuitry whenever the CPU requests to READ memory between 68ØØH and 6FFFH (26624-28671 dec.). If you look at the memory map, you'll see that this represents the entire IO memory space. Thus the keyboard can only be read by reading this range of addresses.

The six column lines are held 'high' by six pull-up resistors as shown in the diagram. Thus if no keys are being

pressed, reading any addresses in this range will have bits Ø-5 set to '1'.However, if a key is pressed, it will link the row line to the column line, and as the row line is connected to an address line, if this address line is at logic level 'Ø', the corresponding data line will be pulled logic low, hence setting that bit to a zero!

Thus keyboard scanning is carried out by taking each of the address lines logic low, and sampling the bits which appear on the data bus. For example, to read the top row of keys, the first row, being address line AØ, must be taken low. If discrete keys are to be read, all other address lines must remain high. If you consider this to be binary 11111111Ø, this means the 8 least significant address lines must contain 254. As IO memory starts at 26624, this represents location 26878 (68FEH). Now if the R key is pressed at that moment, the value returned on the data bus will be 11011111 binary (223 decimal).Similarly, if the 'Q' is pressed, bit 4 will be low,producing 239.

A table showing the row address and corresponding value for each key is shown below. To check for a certain key, simply read the address shown at the left of the letter,and compare the value you get with that shown above the column. Note that due to the simplified decoding, each row effectively occupies several addresses, but the lowest is only shown here.

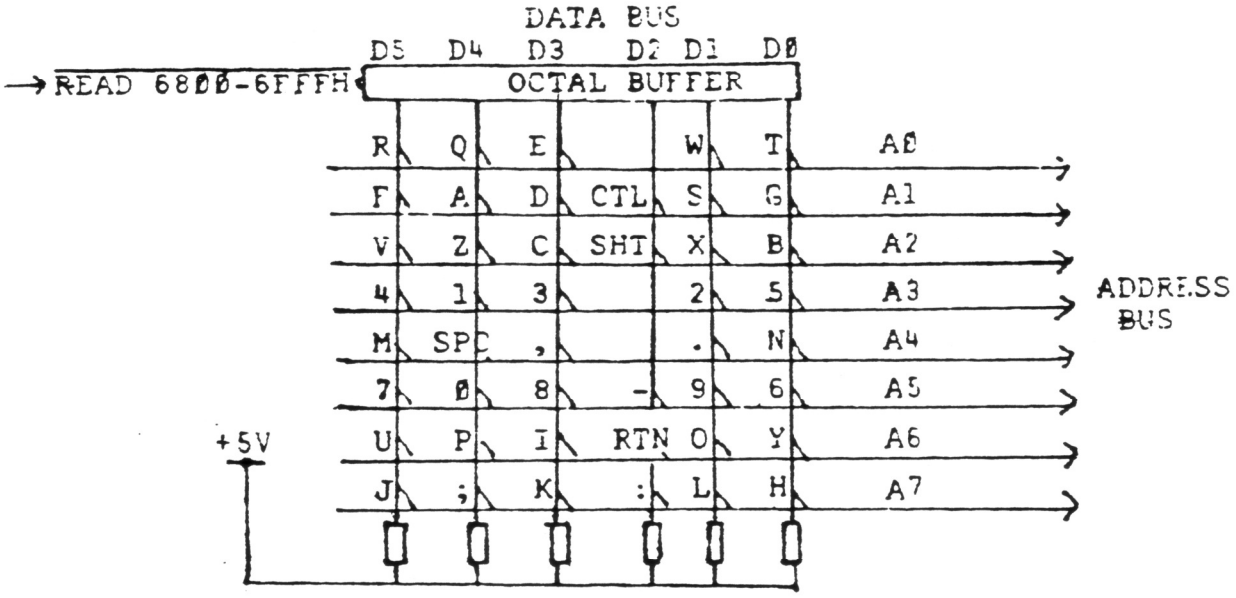
VALUE:	223	239	247	251	253	254
ROW ADDRESS:						
68FEH (26878)	R	Q	E		W	T
68FDH (26877)	F	A	D	CTR	S	G
68FBH (26875)	V	Z	C	SHIFT	X	B
68F7H (26871)	4	1	3		2	5
68EFH (26863)	M	SPC	,		.	N
68DFH (26847)	7	Ø	8	-	9	6
68BFH (26815)	U	P	I	RTN	O	Y
687FH (26751)	J	;	K	:	L	H

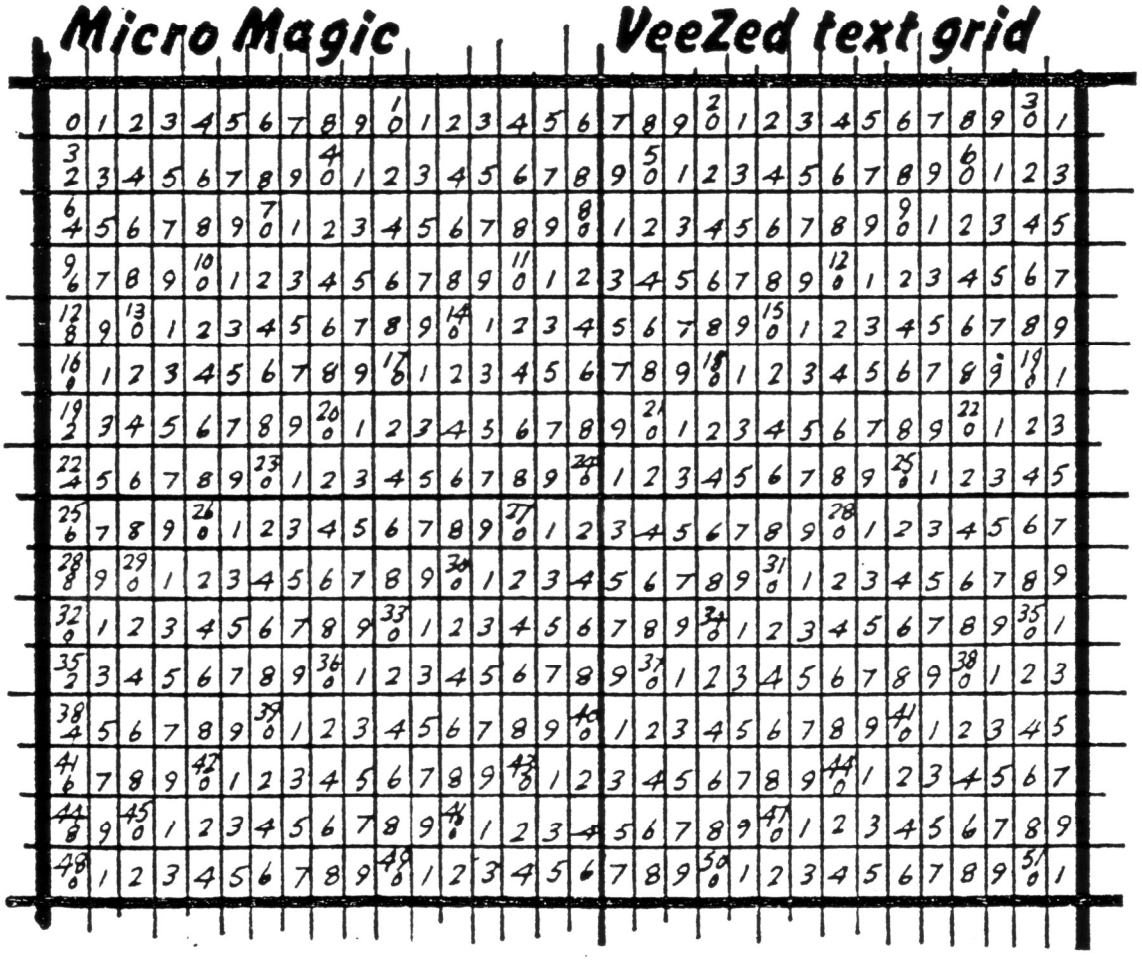
Using the table above you can scan the keyboard in BASIC using the PEEK command.

This is a relatively crued and inefficient way of interfacing a keyboard to the rest of the computer. Larger systems have a keyboard with its own electronics to convert each keypress to an ASCII code.

This method of arranging the keyboard does have its advantages, however.You've probably noticed that there are gaps on the VZ keyboard matrix, not used by any of the keys. These can be used to provide that feature found on many other micro's - function keys.Next month's 'Hardwerk' will present details of how to add three function keys.

One disadvantage of the system is that it's more difficult to add your own keyboard (for VZ200 users), unless you use a VZ300 keyboard (thought to be available at the Service Dept. Dick Smith Electronics). Most commercially-available keyboards have an ASCII output. However, for those really enthusiastic hackers, it is possible to wire your own keyboard matrix. We managed to use a Microbee keyboard, which are readily available and consist of a frame and the keys only.But unless you're an experienced constructor, you're stuck with the VZ300 board at best! Its certainly possible to interface an ASCII keyboard through the IO Port, but to read it would require re-writing the scanning routine in ROM.





LOTTO BY ALAN TOZER

```
130 CLS
135 PRINT"      LOTTO BY ALAN TOZER"
140 REM STORE NUMBERS IN ARRAY
150 DIMB(40)
155 FORI=1TO40
160 B(I)=I
165 NEXT
170 RANGE%=40
180 REM SELECT SIX NUMBERS
190 PRINT"NUMBERS-";
200 FORH=1TO6
210 GOSUB390
220 NEXTH
225 GOSUB500
230 FORI=1TO6
235 PRINTLOT(I);
240 NEXT
330 PRINT:INPUT"DO YOU WANT ANOTHER SET Y/N";ANS$
350 IFANS$="Y"THEN155
360 PRINT"GOOD LUCK"
370 END
380 REMSELECT FROM DECREASING RANGE OF NUMBERS
390 D=RND(RANGE%):D=INT(D+1)
400 LOT(H)=B(D)
410 REMRESET ARRAY DELETING SELECTED NUMBERS
420 F=INT(RANGE%-1)
430 IFD>FTHENB(D)=0:GOTO470
440 FORK=DTO INT(RANGE%-1)
450 B(K)=B(K+1)
460 NEXTK
470 RANGE%=RANGE%-1
480 RETURN
490 LOT(J)=TEMP
500 REMSORT
510 FORJ=1TO6
520 L=J
530 FORK=JTO6
540 IFLOT(L)<LOT(K)THEN560
550 L=K
560 NEXT
570 TEMP=LOT(L)
580 LOT(L)=LOT(J)
590 LOT(J)=TEMP
600 NEXT
610 RETURN
```

P.O.BOX 1972 C.P.O AUCKLAND N.Z

(C) AUCKLAND VZ 300/200 COMPUTER USER'S CLUB 1986

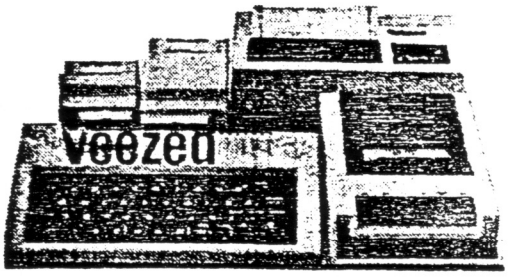
ALL MATERIAL IN THIS NEWSLETTER IS SUBJECT TO COPYRIGHT..
CONTRIBUTED PROGRAMS & ARTICLES CAN BE REPRODUCED ON THE UNDERSTANDING THAT THEY ARE FOR THE PRIVATE USE OF SUBSCRIBERS ONLY.
COPYRIGHT IS ALWAY'S RETAINED BY THE AUTHOR.
THIS CLUB WILL NOT BE HELD RESPONSIBLE FOR THE BREECHING OF COPYRIGHT CAUSED BY THE PUBLICATION OF ANY MATERIAL SENT IN UNDER A FALSE NAME.

YOU ARE WELCOME TO PHONE ME IF YOU HAVE PROBLEMS
PHONE 761-451

Are you interested in writing for our newsletter?

We require members to submit articles, programs, tips and comments to us for publication in this newsletter. It can be ANYTHING you have learnt or are interested in.

MICRO MAGIC



Beginner's Guide to the Veezed ~ 1

AUTO

■ This, the first of 6-issues of MICRO MAGIC begins by introducing a method of AUTOMATIC LINE NUMBERING. In a future issue we will be publishing a feature telling you how to use the very many extra commands that are built in to your computer. The system of using AUTO described here, is similar to that used to acess all the others.

Begin by typing Line 10 in the following manner...

```
10 #100,10[RETURN]
```

Imagine that the HASH SIGN # is the word AUTO.
Now type POKE31469,183:RUN RUN is part of this routine.
As your line 10 is RUN the HASH SIGN has disappeared. So has 100,10.
All you will see is a line number. The command AUTO is in place and line numbers will be generated starting at 100 and will appear like magic in tens everytime you hit RETURN.
Do not add anything to line 10 for fear of overtyping the invisible AUTO 100,10. Should you use BREAK the Automatic Line Numbering will cease. To get them going again you need to type...
POKE30945,175 and AUTO will resume from where it was halted.

Some of the instructions contained in MICRO MAGIC concerning AUTO were written before this easier way was discovered. We advise you stick with this method given here.
When the program is completed line 10 will need to be cancelled.

■ When your Granny was a girl pencil and paper games were very popular. NOUGHTS & CROSSES, BATTLESHIPS & SUBMARINES and more. This, our first MICRO MAGIC program, called BUTCHER was quite a popular pencil and paper game. Sometimes called HANGMAN the object of the game was to guess a secret word letter by letter while for every wrong guess part of the poor old butcher was drawn hanging from the gallows! The player has to guess the word before the ten parts of the whole man has been drawn. The word is represented by a cross for each letter and a clue to the word is given.

■ NOTE: All words and letters underlined should be entered in INVERSE.

BUTCHER

```
100 CLS:CLEAR:DIMB$(12):COLOR3,1:GOSUB620
```

Clear the screen. CLEAR will empty all variables that may have been used in a previous running of this or another program. DIMB$(12) reserves twelve subscripted B$ string variables. In a LOOP B$(1) to B$(12) will be used to store the individual letter of any word containing up to twelve letters. A colour is chosen and then we go to the sub-routine at line 620 to pick up the opening TITLE of this program. We will enter this sub-routine next.

```
620 SOUND1,5:CLS:PRINT"■■■
```

```
150 PRINT@259,"ENTER CLUE";:INPUTZ$:CLS
```

Of course, this is the place to enter the CLUE which should be kept as short as possible. Now enter the following Temporary Line and RUN the Program. If you are using AUTO use RUN100.

```
160 A=LEN(A$):PRINT@257,"▀▀LETTERS▀▀";:PRINTUSING"##";A:COLOR5
```

"A" now contains the number of letters in the GUESSWORD. "LETTERS" has 2 SHIFTJ graphics at either end to add colour. PRINTUSING"##" ensures that no matter whether the GUESSWORD has one digit or two, the number printed will be in line with other numbers soon to be printed on the screen. "A", after the semicolon, prints the number of letters in the GUESSWORD just after the word LETTERS. Then change colour ready for the next line.

```
170 B=A+6:PRINT@289,"▀▀GUESSES▀▀":PRINT@321,▀▀ALLOWED▀▀"B
```

"B" becomes the number of letters in the word plus 6. After printing GUESSES ALLOWED "B" is printed. This means that the player will have six more guesses than there are letters in the GUESSWORD. You may decide to change this after playing the game a few times. Both GUESSES and ALLOWED have 2 SHIFTJs either end.

```
180 PRINT@353,"▀▀ERRORS▀▀":PRINT@385,"▀▀ALLOWED▀▀ 10"
```

The words ERRORS and ALLOWED both have 2 SHIFTJs either end. As there are ten parts to the "HANGING" the errors permitted have been set at 10.

```
190 PRINT@40,Z$
```

PRINT the CLUE.

```
200 PRINT@161,"GUESS COUNT":PRINT@193,"ERROR COUNT"
```

The words GUESS COUNT and ERROR COUNT because of their lengths have no SHIFTJ graphics. Both will have numbers added after them as the game is played.

```
210 FORX=6TOA*2STEP2:PRINT@X,"X":NEXT
```

This LOOP is from screen position 6 to twice the length of the GUESSWORD. STEP2 will print an INVERSE "X" for each letter, two spaces apart.
And this might be a good time to test the program and get everything right before going any further. It is also a good idea to CSAVE what you have already typed to guard against losing the lot through a loss of power or some other catastrophe. But before you TEST RUN or CSAVE add a Temporary Line to prevent the program going through to the sub-routine at lines 620 to 640 after 210.

```
220 GOTO220
```

```
220 FORI=1TOA:B$(I)=MID$(A$,I,1):NEXT
```

FORI=1TOA is a LOOP where each letter of the hidden word is allocated a STRING VARIABLE in turn, as the LOOP progresses. B$(I) becomes B$(1) first and is made the first letter of A$ (which is the hidden word). MID$ selects "I" letter from A$, the "1" ensuring it selects only one letter. As the LOOP progresses B$(2) becomes MID$(A$,2,1) which is the second letter of the hidden WORD. The LOOP continues for the length of the WORD (from 1 to however many letters there are in the word).

```
230 L=448
```
The LOOP for counting the guesses and dealing with each one begins on the next line.
It was Necessary to make "L" =448 outside the playing loop.
It is to be used for printing all the letters used in guesses at the bottom of the screen.
Inside the game loop will be a command to increase "L" for each guess.

```
240 FOR J=1 TO B:PRINT@70,"";:INPUT"GUESS LETTER";X$
```
As "B" is the length of the WORD plus 6 then this LOOP is going to run until the limit of guesses has expired.
To put the INPUT in the right place on the screen we PRINTed at 70,nothing (""), and by using a semicolon we ensured that the INPUT should be in the right place.
So X$ becomes the first guess letter.

```
250 PRINT@L,X$:L=L+2
```
Print the letter (X$) at 448 then increase L by 2 so that the next GUESSLETTER will be Printed two spaces from this one.

```
260 P=6:FORI=1TOA
```
Here we go with another LOOP! As this LOOP is INSIDE the game loop, beginning at line 240, this is what is called a NESTED LOOP. This one is for Printing CORRECT guesses on top of those INVERSE Xs at the top of the screen. We start by making P equal 6, which is the location of the first X. The LOOP is to run for the length of the chosen word.

```
270 IFX$=B$(I)THENPRINT@P,X$
280 IFX$=B$(I)THENC=1
290 IFX$=B$(I)THENR=R+1
```
Here we have three IFs to Pick out and deal with correct guesses. Line 270 says: if X$ is equal to B$(I),which is firstly B$(1) and contains the first letter of the chosen word then Print at 6 the correctly guessed letter.
If the GUESSLETTER does not equal the first letter of the word then the computer ignores this line. When the LOOP comes round again it will check the GUESSLETTER with the second letter of the WORD, Z$(2), and so on, "P" will have to be increased each time so that correct guesses are Printed in the right Places.
Line 280 again tests the GUESSLETTER against each letter in turn of the chosen word. But this time it makes the variable C a 1 if the guess is correct.
NOTE: C is never more than 1. Even if the GUESSLETTER is in the hidden word more than once, C remains at 1. Only if the GUESSLETTER is WRONG then C will be 0.
Line 290 is similar to line 280 except that R is increased by 1 for every correct guess.

```
300 IFR=ATHEN400
```
When R (in line 290) reaches the same number as there are letters in the word then we know that the Player has guessed the whole word. If the word, for instance, had been MISSISSIPPI, although there are 11 letters, R would have reached 11 in 4 guesses if all the guesses were correct. With a CLUE like "A LONG RIVER" it would be quite likely. So, when R does reach the same number as there are letters in the word then we must jump out of the game loop and end the game. This is where line 400 comes in.

```
400 CLS:PRINT@197,"YOU'VE GOT IT!"A$:GOTO610
```
Clear the screen - Print the message - then go to line 610.

```
610 FORI=1TO3000:NEXT:RUN
```
This is just a Pause to allow the Player to read the message and RUN starts a new game.
Now back to line 310.

```
310 P=P+2:NEXT
```
P (which is the screen Positon of the INVERSE Xs) is now increased by 2 ready for Printing the next guess at 8 if correct. NEXT is the end of the NESTED LOOP which was for checking GUESSLETTERS against each letter of the word. The NEXT for the Game LOOP is yet to come. We still have to deal with CORRECT and WRONG guesses. We haven't hung anybody yet!

```
320 IFC=0THEND=D+1
330 IFC=1THENPRINT@225,"■ RIGHT!■"ELSEPRINT@225,"■ WRONG!■"
340 IFC=0THENGOSUB410
```
Another three IFs. This time to make use of C.
Line 320 "IF C=0" means: if the GUESSLETTER was WRONG, (remember, C only becomes a 1 if the guess is CORRECT and stays 0 if the guess was WRONG), then D equals itself Plus 1. We are counting the number of correct guesses. Even if a GUESSLETTER appears more than once in the word it is still counted as one correct guess. We will be using C shortly.
Line 330 says: if the guess was CORRECT then Print "RIGHT!" but if the guess was WRONG then Print "WRONG!".
NOTE: The words "RIGHT!" and "WRONG!" both have 2 SHIFTJ graphics at either end, and are in INVERSE starting with an INVERSE space.
Line 340 says: if the guess was WRONG then go to the sub-routine at 410 and come back on the command RETURN.

We are not going to type in the line 410 sub-routine yet as, from line 420 onwards, are all the HANGING graphics, and it will be easier to do all these together. They are the GALLOWS, HEAD, BODY and so on. We only need about five more lines to finish the actual Playing of the game. Then we can TEST RUN and cure any BUGS before dealing with the hanging of the BUTCHER.

```
350 C=0:PRINT@205,D
```
Here we return C to zero ready for the next guess, and D (which is?) - yes, that's right - D is the number of correct guesses, is Printed on the screen at Position 205.

```
360 PRINT@173,J
```
J (which is the game LOOP) is the total number of guesses or the number times the LOOP has cycled. Print J at 173. It will be updated as the game Progresses.

```
370 NEXT
```
...we've finally arrived at the end of the game LOOP!
Many versions of BASIC would insist on NEXTJ, but the VeeZed does not.
When the computer reaches this line it means that all the Permitted guesses have been used up, and the Player has not guessed the word. The Program goes on to the next line, which is....

```
380 CLS:PRINT@225,"SORRY!":PRINT" YOU'VE RUN OUT OF GUESSES."
```

```
390 PRINT:PRINT" THE WORD WAS "A$:GOTO610
```

Line 380 clears the screen and prints the message.
Line 390 says what the WORD was, and then goes to 610, which is already typed in. It is the 3 second pause followed by RUN to start another game.

■ There's enough program now to TEST RUN a game. But, before you do that you will need 2 Temporary Lines. You cannot use 400 because that is the line that says "YOU'VE GOT IT!", so, why not just use 392.

```
392 GOTO392
410 RETURN
```

Line 410 takes care of the GOSUB410 in line 340.

In many programs where POKEs are used, the TEST RUNNING of a program can cause a computer to CRASH! An error can send the computer into hysterics where the only cure is to switch off and on again. There is very little chance of this program CRASHING but it is still a good idea to CSAVE all that typing.

With the Temporary lines in place RUN the program.
If you are still using AUTO then type RUN100.
When you have corrected any errors and the program is running OK,CSAVE it again.Now all that remains to be added is All the penalties for wrong guesses.

Ready to start again...

Delete Line 392.
Start AUTO at 410 - INCR 10.

In the next three lines we are going to use the SHORTFORM for "THEN". Instead of the command THEN we type a COMMA. This is exactly the same as THEN but takes less room in a program line.

```
410 IFD=1,440ELSEIFD=2,480ELSEIFD=3,490ELSEIFD=4,510
420 IFD=5,520ELSEIFD=6,530ELSEIFD=7,540ELSEIFD=8,550
430 IFD=9,560ELSEIFD=10,570
```

Notice that the computer must have an IF for each of the statements that are linked together with ELSEs. Now you know why the keyboard has THEN, ELSE and IF close together. While we are using a SHORTFORM for THEN, it can be mentioned that there is a SHORTFORM for PRINT. It is a QUESTION MARK. Strangely, when ? is used for PRINT it appears, when the program is LISTed, as the word PRINT. It can be used to squeeze a few extra characters into a line.

Lines 410 to 430 are saying: if the number of WRONG guesses is 1 then go to 440, but if the number of WRONG guesses are 2 then go to 480, etc. etc. The first WRONG guess will print the GALLOWS, the second WRONG guess will print the man's HEAD, and so on.
Remember that these three lines are part of a GOSUB and a RETURN will send the program back to the command following the GOSUB. The line containing the GOSUB is line 340, which is INSIDE the game loop.

Following are the lines that actually draw the HANGING man. There are ten parts and each part ends with a RETURN. To let you know what graphic characters to use we have added a code, (in brackets). Do not type these in. (12J) would simply mean... use 12 SHIFTJ graphic characters.

If you have been using AUTO then delete it now.

To delete AUTO simply type the AUTO line numbers one at a time each followed by RETURN.

As we proceed with the GRAPHICS, AUTO would interfere with the method of checking each part of your graphic picture as you go.

Each time you enter RETURN put an APOSTROPHE in front of the word, like 'RETURN.
This is a SHORTFORM of REM and will cause the computer to ignore the RETURN.
If you then enter the following Temporary Line: 599 GOTO599 you will be able to check each bit of graphics as they are entered by typing CLS:RUN.
After checking RUBOUT the APOSTROPHE before continuing.

Here we go..... Delete line 599

THE CODE IN BRACKETS AT THE END OF LINES IS TO LET YOU KNOW HOW MANY AND WHAT SORT OF GRAPHICS TO USE. (14J) MEANS USE 14 SHIFT J GRAPHICS). DO NOT TYPE THIS CODE.

```
440 COLOR8:PRINT@113,"▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀"          (14J)
450 FORI=157TO413STEP32
460 PRINT@I,"▀▀":NEXT                              (2J)
470 PRINT@433,"▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀":RETURN            (1J+8T+5J)
480 COLOR5:PRINT@148,"▀▀▀":PRINT@180,"▀▄▄▀":RETURN  (1Q+1J+1W & 1Y+1J+1Y)
490 COLOR5:PRINT@213,"▬":COLOR6:PRINT@243,"▚▚ ▚▚"   (1Y & 2H+1S+2H)
500 PRINT@275,"▚▚▚▚▚":RETURN                       (5H)
510 COLOR6:PRINT@248,"▚▚":COLOR5:PRINT@250,"▀▀"    (2H & 1I+@)
520 COLOR5:PRINT@239,"@▀":COLOR6:PRINT@241,"▚▚":RETURN  (@+1U & 2H)
530 PRINT@308,"▪▪▪":COLOR7:PRINT@340,"▀▀▀":RETURN  (3D & 3J)
540 PRINT@372,"▀":PRINT@404,"▀":RETURN             (1J & 1J)
550 PRINT@374,"▀":PRINT@406,"▀":RETURN             (1J & 1J)
560 COLOR4:PRINT@434,"▄▄":RETURN                   (1J & 1Y)
570 PRINT@433,"          "                         (10 SPACES)
580 PRINT@434,"▄▄ ▄▄":SOUND1,6                     (2J+1Y+SPACE+1Y+1J)
```

There are now only two more line to enter.
Line 590 will clear a space on the screen in which to print the secret word.

```
590 N=129:FORI=1TO10:PRINT@N," ":N=N+1:NEXT:PRINT@129,A$
```

N is the screen location for the LOOP to print nothing!
By nothing we mean empty quotes, " " and as N is increased by 1 each time the next "nothing" to be printed will be one space further on each time. When ten spaces have been cleared, we then print in that space the secret word.

```
600 PRINT@151,"EE-EEK!":SOUND31,9
```

The program is now complete.

As you will appreciate, this MICRO MAGIC program has been checked for correctness. However, experience tells us, that the VeeZed Gremlin may have hidden an error which escaped detection. We apologise if this is the case and hope that you will be able to put it right and let us know what was wrong.